IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: **John Anthony Beaven et al.**	§	Attorney Docket No. **GB920000055US1**
Serial No.: **09/808,501**	§	Examiner: **Insun Kang**
Filed: **March 14, 2001**	§	Art Unit: **2193**
For: **METHOD, SYSTEM AND COMPUTER PROGRAM FOR DERIVING AND APPLYING QUALITY OF SERVICE SPECIFICATIONS IN A COMPONENT-BASED DEVELOPMENT ENVIRONMENT**	§	Confirmation No.: **3614**

SECOND SUPPLEMENTAL APPEAL BRIEF UNDER 37 C.F.R. 41.37

Mail Stop Appeal Briefs - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Second Supplemental Appeal Brief is submitted in support of the Notice of Appeal filed November 20, 2008, in response to the final rejection of claims 1-49 in the above-identified application, and in response to the Notice of Non-Compliant Appeal Brief of June 15, 2009. Please charge a one-month extension of time fee to International Business Machines Corporation Deposit Account No. **09-0461**.

REAL PARTY IN INTEREST

The real party in interest in the present Appeal is International Business Machines Corporation, the Assignee of the present application.

RELATED APPEALS AND INTERFERENCES

No appeals, interferences, or judicial proceedings are known to Appellants, the Appellants' legal representative, or Assignee, which may be related to, directly affect, or would be directly affected by or have a bearing on the Board's decision in the pending Appeal.

STATUS OF CLAIMS

Claims 1-14, 17-30, 33-46, and 49 were rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent Application Publication No. 2002/0059079 (hereinafter "Negri") in view of "QoS for Distributed Object Computing Middleware – Fact or Fiction?" (hereinafter "Schmidt"). Claims 15, 16, 31, 32, 47, and 48 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Negri in view of Schmidt and "Quality of Service Aware Distributed Object Systems" (hereinafter "Koistinen"). The rejection of claims 1-49 is appealed.

STATUS OF AMENDMENTS

No amendments to the claims have been proposed or entered subsequent to the Final Office Action that led to this appeal.

SUMMARY OF CLAIMED SUBJECT MATTER

With reference to Figs. 2 and 4 (see page 12, line 1 through page 13, line 17 and page 16, lines 10-22), independent claim 1 is directed to a computer system (see page 16, lines 24-25) having a processor configured for component-based processing. The system includes a component specification element that specifies components, which are reusable components (212, 214) (see page 12, lines 11-12). The system further includes a control flow specification element that specifies control flows (216), a data flow specification element that specifies data flows (218), and a resource specification element that specifies resources (213) (see page 12, lines 12-14). The system also includes a quality of service specification derivation element (220) (see page 12, lines 14-19). The quality of service specification derivation element (220) having for output an application model in combination with a quality of service specification derived by implication

from relations between the components (212, 214), the control flows (216), the data flows (218), and the resources (213). The quality of service specification derivation element (220) tests the components (212, 214) and the relations between the components (212, 214) to derive the quality of service specification (see page 12, lines 14-19). The quality of service specification is made available to a runtime engine (250) for deployment as a runtime contract (222) in a runtime processing environment (see page 12, line 19 through page 13, line 1 and page 16, lines 10-14). As is set forth at page 15, lines 14-19, the runtime contract (222) is defined to specify services that are required to be provided by a target environment.

Dependent claim 8 furthers limits the runtime contract of independent claim 1 to comprising a completion requirement contract that specifies transactional behavior (see, for example, page 7, lines 13-14).

With reference to Figs. 2 and 4 (see page 12, line 1 through page 13, line 17 and page 16, lines 10-22), independent claim 18 is directed to a method for component-based processing that includes specifying components (which are reusable components (212, 214) (see page 12, lines 11-12)), specifying control flows (216), specifying data flows (218), specifying resources (213) (see page 12, lines 12-14), and deriving a quality of service specification by implication from relations between the components (212, 214), the control flows (216), the data flows (218), and the resources (213). A quality of service specification derivation element (220) is employed to test the components (212, 214) and the relations between the components (212, 214) to derive the quality of service specification (see page 12, lines 14-19). The quality of service specification is made available to a runtime engine (250) for deployment as a runtime contract (222) in a runtime processing environment (see page 12, line 19 through page 13, line 1 and page 16, lines 10-14). As is set forth at page 15, lines 14-19, the runtime contract (222) is defined to specify services that are required to be provided by a target environment.

Dependent claim 24 furthers limits the runtime contract of independent claim 18 to comprising a completion requirement contract that specifies transactional behavior (see, for example, page 7, lines 13-14) .

With reference to Figs. 2 and 4 (see page 12, line 1 through page 13, line 17 and page 16, lines 10-22), independent claim 34 is directed to a computer program product (see page 16, line 24 through page 17, line 10) comprising computer program code on a computer readable storage medium to, when loaded and executed on a computer, cause the computer to specify components (which are reusable components (212, 214) (see page 12, lines 11-12)), specify control flows

(216), specify data flows (218), specify resources (213) (see page 12, lines 12-14), and derive a quality of service specification by implication from relations between the components (212, 214), the control flows (216), the data flows (218), and the resources (213) (see page 12, lines 14-19). A quality of service specification derivation element (220) is employed to test the components (212, 214) and the relations between the components (212, 214) to derive the quality of service specification (see page 12, lines 14-19). The quality of service specification is made available to a runtime engine (250) for deployment as a runtime contract (222) in a runtime processing environment (see page 12, line 19 through page 13, line 1 and page 16, lines 10-14). As is set forth at page 15, lines 14-19, the runtime contract (222) is defined to specify services that are required to be provided by a target environment.

Dependent claim 40 furthers limits the runtime contract of independent claim 34 to comprising a completion requirement contract that specifies transactional behavior (see, for example, page 7, lines 13-14) .

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1-14, 17-30, 33-46, and 49 are patentable under 35 U.S.C. § 103(a) over Negri in view of Schmidt.

Whether claims 15, 16, 31, 32, 47, and 48 are patentable under 35 U.S.C. § 103(a) over Negri in view of Schmidt and Koistinen.

ARGUMENT

REJECTIONS UNDER 35 U.S.C. § 103(a)

At page 2 of the Final Office Action, claims 1-14, 17-30, 33-46, and 49 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Negri and Schmidt. At page 9 of the Final Office Action, claims 15, 16, 31, 32, 47, and 48 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Negri in view of Schmidt and Koistinen. The rejection of the claims under 35 U.S.C. § 103(a) is not well founded and should be reversed for at least the reason that the combination of Negri and Schmidt (as well as Koistinen) does not teach or suggest all of the features set forth in Appellants' independent claims 1, 18, and 34. To establish *prima facie*

obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. *In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974). "All words in a claim must be considered in judging the patentability of that claim against the prior art." In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970). If an independent claim is nonobvious under 35 U.S.C. § 103, then any claim depending therefrom is nonobvious. *In re Fines*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988).

Independent Claims 1, 18, and 34

As noted in Appellants' specification (see, for example, page 5, lines 1-4), quality of service specifications have traditionally been encoded within applications, application components, or communication flow elements of a system. In contrast, according to an aspect of Appellants' claimed subject matter, a quality of service specification is made available to a runtime engine for deployment as a runtime contract (i.e., a contract for service). A quality of service specification derivation element is employed to test components and relations between the components to derive the quality of service specification, which is also derived by implication from the relations between the components, control flows, data flows, and resources.

With specific reference to the rejection of Appellants' independent claims 1, 18, and 34 in view of Negri, Appellants note that Negri is directed to deriving an e-service management strategy based on a business process model. More specifically, Negri discloses utilizing a business process specification (that describes a business process of an e-service) and an infrastructure specification (that describes an infrastructure that supports the e-service) to derive an e-service management strategy for the infrastructure that provides a desired quality of service for the e-service (see, for example, Abstract). More to the point, Negri is directed to providing a desired quality of service for an e-service (see, for example, paragraphs [0013] and [0040]). In this endeavor, Negri discloses employing business experts (BeXs), which correspond to a distributed intelligence technology that is deployed on components included in a system. As is disclosed, a BeX is developed independent of a specific model and utilizes relationships within an e-service model to analyze the impact of related components, and includes a local processing engine (see, for example, paragraph [0057]). As is noted at paragraph [0058], a BeX is in essence a recorded best practice for a component of a service that can be implemented without on-site development. By analyzing components locally, only a relatively small number of meaningful events and supporting data is required to be provided to an e-service management

server (see, for example, paragraph [0059]).

In contrast, Appellants' claimed subject matter is directed to employing a quality of service specification derivation element that provides a quality of service specification that is derived by implication from relations between components, control flows, data flows, and resources. The quality of service specification derivation element also test the components and the relations between the components to derive the quality of service specification, which is made available to a runtime engine for deployment as a runtime contract in a runtime processing environment. As is set forth at page 15, lines 14-19 of Appellants' specification, the runtime contract specifies services that are required to be provided by a target environment.

Appellants agree that Negri does not disclose that a quality of service specification is made available to a runtime engine for deployment as a runtime contract in a runtime processing environment (see page 4 of the Final Office Action). However, Appellants do not agree that it would have been obvious to modify Negri's disclosed system to derive a runtime contract from a service delivery model. Moreover, among other deficiencies, Negri does not teach (or suggest) the derivation of a runtime contract from a quality of service specification (that is derived by a quality of service specification derivation element that tests components and relations between the components) or for that matter the use of runtime contracts. Furthermore, Appellants note that the stated rationale for declaring Appellants' claimed subject matter obvious (i.e., that "[t]he service delivery model in Negri would help 'ensuring the quality of eService delivery (0023)' when deployed at runtime"), while convoluted, appears to be based solely on hindsight in view of Appellants' own disclosure as the Final Office Action failed to cite any prior art reference in support of the position.

Additionally, while Negri paragraphs [0056] and [0060] disclose the use of BeXs, the Negri BeXs do not test components and relations between components to derive a quality of service specification. As is noted above, while the Negri BeXs record best practices for a component of a service, the BeXs provide only a relatively small number of meaningful events and supporting data to an e-service management server (see, for example, paragraph [0059]) to facilitate control of a quality of service for an e-service and do not create a quality of service specification. As such, Negri does not teach or suggest a quality of service specification derivation element that test components and relations between components to derive a quality of service specification.

Nor does Negri (claim 1, paragraphs [0036], [0050], [0057], and/or [0063]) teach or

suggest a quality of service specification derivation element that also derives a quality of service specification by implication from relations between components, control flows, data flows, and resources. Appellants note that Negri claim 1 is merely directed to deriving an e-service management strategy to ensure a service quality of the e-service by incorporating needs imposed by a business process in criteria for managing an infrastructure and monitoring an impact of the infrastructure in accordance with the business process. Appellants further note that Negri paragraph [0036] is merely directed to a service level management (SLM) tool that measures service delivery. Appellants also note that Negri paragraph [0050] merely discloses that an e-service model establishes implicit and explicit relationships between components (that can depend on each other, share common resources, exchange data with each other, collect common statistics, and work together in complex flows of control) that defines an actual topology of the model. Appellants also note that Negri paragraph [0057] is merely directed to the properties of a BeX, which are described above. Finally, Appellants note that Negri paragraph [0063] merely provides an overview of the functionality of a BeX.

For at least the reasons set forth above, Appellants respectfully submit that Appellants' independent claims 1, 18, and 34 are allowable over the applied art of record.

Dependent Claims 8, 24, and 40

Dependent claims 8, 24, and 40 further limit their respective independent claims to defining a runtime contract as comprising a completion requirement contract that specifies transactional behavior. In rejecting claim 8, the Final Office Action (at page 7) stated "Negri does not explicitly disclose that said runtime contract comprise a completion requirement contract." Appellants agree that Negri does not teach or suggest a runtime contract that is a completion requirement contract and further submit that Negri does not teach or suggest the use of runtime contracts. In rejecting claim 8, the Final Office Action (at page 7) further stated "Negri discloses a service level agreement (SLA) that specify the levels of service (0014) and a Service level management tool that measures service delivery (0036). Therefore, it would have been obvious for a person having ordinary skill in the pertinent art at the time of the invention was made to modify Negri's disclosed system to include a completion requirement contract specifying transactional behavior in the service delivery contract to help 'ensuring the quality of eService delivery (0023).'"

Appellants agree than an SLA is an agreement between parties that specifies a level of

service to be provided and agree that a service level management tool may be used to measure delivery of a service. However, Appellants do not agree that an SLA and a service management tool render obvious a runtime contract in the form of a completion requirement contract that specifies transactional behavior. In sum, Negri (alone or in combination with Schmidt) does not teach or suggest a runtime contract that takes the form of a completion requirement contract that specifies transactional behavior. Nor does Negri teach or suggest a runtime contract that takes the form of a completion requirement contract. For at the above-reasons, dependent claims 8, 24, and 40 are allowable over the applied art.

Additionally, Appellants respectfully submit that dependent claims 2-17, 19-33, and 35-49 are also allowable for at least the reason that the claims depend on allowable claims.

CONCLUSION

The foregoing remarks demonstrate that neither Negri nor Schmidt, alone or in combination, teach or suggest each feature of independent claims 1, 18, and 34 as required to support a rejection under 35 U.S.C. § 103(a). Accordingly, Appellants submit that the rejection of claims 1-49 is in error and should be overturned. No additional fees are believed to be required. If, however, any additional fees are required, please charge those fees to International Business Machines Corporation Deposit Account No. **09-0461**.

Respectfully submitted,

Michael R. Long
Reg. No. 42,808
DILLON & YUDELL LLP
8911 N. Capital of Texas Hwy., Suite 2110
Austin, Texas 78759
(512) 343-6116
ATTORNEY FOR APPELLANTS

CLAIMS APPENDIX

1. A computer system having a processor configured for component-based processing, said system comprising:

a component specification element that specifies components, wherein the components are reusable components;

a control flow specification element that specifies control flows;

a data flow specification element that specifies data flows;

a resource specification element that specifies resources; and

a quality of service specification derivation element, the quality of service specification derivation element having for output an application model in combination with a quality of service specification derived by implication from relations between the components, the control flows, the data flows and the resources;

wherein the quality of service specification derivation element tests the components and the relations between the components to derive the quality of service specification;

and wherein said quality of service specification is made available to a runtime engine for deployment as a runtime contract in a runtime processing environment.

2. A computer system as claimed in claim 1, further comprising a runtime engine for deploying said runtime contract.

3. A computer system as claimed in claim 1, wherein said runtime contract comprises a messaging requirement contract.

4. A computer system as claimed in claim 1, wherein said runtime contract comprises a transactionality requirement contract.

5. A computer system as claimed in claim 1, wherein said runtime contract comprises a security requirement contract.

6. A computer system as claimed in claim 1, wherein said runtime contract comprises a recoverability requirement contract.

7. A computer system as claimed in claim 1, wherein said runtime contract comprises a completion requirement contract.

8. A computer system as claimed in claim 7, wherein said runtime contract comprises a completion requirement contract specifying transactional behavior.

9. A computer system as claimed in claim 7, wherein said runtime contract comprises a completion requirement contract specifying compensation behavior.

10. A computer system as claimed in claim 1, wherein said runtime contract comprises at least one of a reliability, availability and serviceability requirement contract.

11. A computer system as claimed in claim 1, wherein said runtime contract comprises a quality of delivery requirement contract.

12. A computer system as claimed in claim 1, wherein said runtime contract comprises at least one of a priority requirement and a response goal requirement contract.

13. A computer system as claimed in claim 1, wherein said runtime contract comprises a performance requirement contract.

14. A computer system as claimed in claim 1, wherein said quality of service specification is stored in a repository.

15. A computer system as claimed in claim 1, wherein said quality of service specification is stored in a tagged markup language.

16. A computer system as claimed in claim 15, wherein said tagged markup language is XML.

17. A computer system as claimed in claim 1, wherein said quality of service specification is stored in a modeling language.

18. A method for component-based processing, said method comprising the steps of:

specifying components, wherein the components are reusable components;

specifying control flows;

specifying data flows;

specifying resources; and

deriving a quality of service specification by implication from relations between the components, the control flows, the data flows and the resources;

wherein a quality of service specification derivation element is employed to test the components and the relations between the components to derive the quality of service specification;

and wherein said quality of service specification is made available to a runtime engine for deployment as a runtime contract in a runtime processing environment.

19. A method as claimed in claim 18, further comprising the step of deploying said runtime contract by a runtime engine.

20. A method as claimed in claim 18, wherein said runtime contract comprises a transactionality requirement contract.

21. A method as claimed in claim 18, wherein said runtime contract comprises a security requirement contract.

22. A method as claimed in claim 18, wherein said runtime contract comprises a recoverability requirement contract.

23. A method as claimed in claim 18, wherein said runtime contract comprises a completion requirement contract.

24. A method as claimed in claim 23, wherein said runtime contract comprises a completion requirement contract specifying transactional behavior.

25. A method as claimed in claim 23, wherein said runtime contract comprises a completion requirement contract specifying compensation behavior.

26. A method as claimed in claim 18, wherein said runtime contract comprises at least one of a reliability, availability and serviceability requirement contract.

27. A method as claimed in claim 18, wherein said runtime contract comprises a quality of delivery requirement contract.

28. A method as claimed in claim 18, wherein said runtime contract comprises at least one of a priority requirement and a response goal requirement contract.

29. A method as claimed in claim 18, wherein said runtime contract comprises a performance requirement contract.

30. A method as claimed in claim 18, wherein said quality of service specification is stored in a repository.

31. A method as claimed in claim 18, wherein said quality of service specification is stored in a tagged markup language.

32. A method as claimed in claim 31, wherein said tagged markup language is XML.

33. A method as claimed in claim 18, wherein said quality of service specification is stored in a modeling language.

34. A computer program product comprising computer program code on a computer readable storage medium to, when loaded and executed on a computer, cause said computer to perform the steps of:

specifying components, wherein the components are reusable components;

specifying control flows;

specifying data flows;

specifying resources; and

deriving a quality of service specification by implication from relations between the components, the control flows, the data flows and the resources;

wherein a quality of service specification derivation element is employed to test the components and the relations between the components to derive the quality of service specification;

and wherein said quality of service specification is made available to a runtime engine for deployment as a runtime contract in a runtime processing environment.

35. The computer program product of claim 34 further comprising additional computer program code to, when loaded and executed on a computer, cause said computer to perform the step of:

deploying said runtime contract using a runtime engine.

36. The computer program product of claim 34, wherein said runtime contract comprises a transactionality requirement contract.

37. The computer program product of claim 34, wherein said runtime contract comprises a security requirement contract.

38. The computer program product of claim 34, wherein said runtime contract comprises a recoverability requirement contract.

39. The computer program product of claim 34, wherein said runtime contract comprises a completion requirement contract.

40. The computer program product of claim 39, wherein said completion requirement contract specifies transactional behavior.

41. The computer program product of claim 39, wherein said completion requirement contract specifies compensation behavior.

42. The computer program product of claim 34, wherein said runtime contract comprises at least one of a reliability, availability, and serviceability contract.

43. The computer program product of claim 34, wherein said runtime contract comprises a quality of delivery requirement contract.

44. The computer program product of claim 34, wherein said runtime contract comprises at least one of a priority requirement and a response goal requirement contract.

45. The computer program product of claim 34, wherein said runtime contract comprises a performance requirement contract.

46. The computer program product of claim 34, wherein said quality of service specification is stored in a repository.

47. The computer program product of claim 34, wherein said quality of service specification is stored in a tagged markup language.

48. The computer program product of claim 47, wherein said tagged markup language is XML.

49. The computer program product of claim 34, wherein said quality of service specification is stored in a modeling language.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.